Li et al. EURASIP Journal on Embedded Systems (2017) 2017:5
DOI 10.1186/s13639-016-0038-6

EURASIP Journal on Embedded Systems

REVIEW Open Access

Feedback recurrent neural network-based embedded vector and its application in topic model

Lian-sheng Li[1], Sheng-jiang Gan[2*] and Xiang-dong Yin[1]

Abstract

While mining topics in a document collection, in order to capture the relationships between words and further improve the effectiveness of discovered topics, this paper proposed a feedback recurrent neural network-based topic model. We represented each word as a one-hot vector and embedded each document into a low-dimensional vector space. During the process of document embedding, we applied the long short-term memory method to capture the backward relationships between words and proposed a feedback recurrent neural network to capture the forward relationships between words. In the topic model, we used the original and muted document pairs as positive samples and the original and random document pairs as negative samples to train the model. The experiments show that the proposed model consumes not only lower running time and memory but also has better effectiveness during topic analysis.

Keywords: Wireless sensor networks, Data aggregation, Aggregation tree, Aggregation delay

1 Review

1.1 Introduction

Natural language processing (NLP) [1] is an interdiscipline about linguistics, statistics, computer science, artificial intelligence, and so on. Statistical theory is one of the most important tools in analyzing natural language documents. In a document collection, each term (word or phrase) is denoted with a one-hot vector, and each sentence, paragraph, or document is denoted with a term frequency vector. In a term frequency vector, each element represents the number of occurrence for the corresponding term. With these term frequency vectors, researchers could compute the similarities between documents and thus discover underlying topics in such a document collection [2]. Topic models can be classified into statistical semantic models [3–10] and embedded vector models [11–13]. While capturing the semantics of documents, statistical semantic model computes the similarities between documents with co-occurrence matrix of terms, and embedded vector model uses neighbor(s) to represent the meaning of a target term; however, both of them cannot describe the term orders in a document.

In order to better capture the order relationships between terms, and thus improve the topic discovering effectiveness, this paper classified the order relationships between terms into forward dependence and backward dependence and proposed a feedback recurrent neural network-based topic model. While capturing backward dependences, this paper denoted each term with a one-hot vector and applied LSTM recurrent neural network [14] to compute its corresponding embedded vector. While capturing forward dependences, this paper designed a feedback mechanism for the recurrent neural network.

1.1.1 Related works

Statistical semantic models, such as latent semantic analysis (LSA) [3], probabilistic latent semantic analysis (PLSA) [4], and latent Dirichlet allocation (LDA) [5], are power tools for mining underlying topics in a document collection. PLSA is the probabilistic version of LSA, and both of them compute the similarities between documents with co-occurrences of terms, such as TF-IDF [6], but they ignore the order relationships between terms. For two sentences with synonyms, both LSA and PLSA

* Correspondence: gsj1102@126.com
[2]School of computer sciences Chengdu Normal University, Sichuan, Chengdu 611130, China
Full list of author information is available at the end of the article

consider them as unsimilar, but LDA can capture the semantics behind terms, and consider them as similar sentences. The LDA model assumes that a document collection is generated by distribution of topics, and each topic is a probabilistic distribution of terms. Based on LDA model, researchers also propose HDP-LDA [7], ADM-LDA [8], Tri-LDA [9], MB-LDA [10], and so on. Although the LDA and its extended models take the semantics between terms into consideration, they ignore the term orders in a document. For example, "Tom told Jim" and "Jim told Tom" have different semantics, but the LDA model can never differentiate them.

In recent years, embedded vector models, such as Word2vec [11], GloVe [12], and DSSM [13], are more and more popular, and deep learning for NLP has attracted more and more researchers. By embedding term frequency vectors into continuous multi-dimension vector space (dimension of continuous vector space is much smaller than that of one-hot vector), the similarities between documents can be accurately and quickly computed. During the generating of embedded vectors, Word2vec uses one of the neighbors of the target term, and GloVe uses the average of all neighbors of the target term to represent the target term. In addition, DSSM denotes a document with a set of triple characters, and this tri-character set can only capture the relationships between two adjacent terms.

In order to better capture the order relationships between terms, and thus improve the topic discovering effectiveness, this paper classifies the order relationships between terms into forward dependence and backward dependence and proposes a feedback recurrent neural network-based topic model.

1.1.2 Feedback recurrent neural network-based embedded vectors

In this section, we embed documents into a continuous vector space. Firstly, we generate an embedded vector for each document with simple recurrent neural network, then capture backward dependences of words by adding memory to each neural cell, and finally capture forward dependences by introducing feedback links.

1.1.2.1 Generating embedded vectors with recurrent neural network With recurrent relationships, recurrent neural networks can model the relationship between the current word with its previous one. Theoretically, recurrent neural networks can be used to model sequences of any length.

Given a document collection D, after removing stop words, there are total T different words remained, so each word w_t $(1 \le t \le T)$ can be denoted with a one-hot vector x_t. In this paper, we use a four-latent-layer recurrent neural network to compute the embedded vector for each document, and the structure of the proposed neural network is in Fig. 1. In Fig. 1, the arrows denote the dependences of data between neural cells; x_t and y_t are input and output of the t-th words respectively, and the dimension of latent layers is much smaller than the dimension of the input one-hot vector.

We apply Sigmoid function $\sigma(x) = (1 + \exp(-x))^{-1}$ as the activation function for each neural cell, where $\sigma(x)$ is element-wise. While using the Sigmoid function, data of latent and output layers can be computed as follows.

$$\boldsymbol{h}_t^1 = \frac{1}{1 + \exp\left(-\boldsymbol{U}^1 \boldsymbol{h}_{t-1}^1 - \boldsymbol{W}^1 \boldsymbol{x}_t\right)} \tag{1}$$

$$\boldsymbol{h}_t^i = \frac{1}{1 + \exp\left(-\boldsymbol{U}^i \boldsymbol{h}_{t-1}^i - \boldsymbol{W}^i \boldsymbol{h}_t^{i-1}\right)}, i = 2, 3, 4 \tag{2}$$

$$\boldsymbol{y}_t = \frac{1}{1 + \exp\left(-\boldsymbol{U}^4 \boldsymbol{h}_{t-1}^4 - \boldsymbol{W}^5 \boldsymbol{h}_t^4\right)} \tag{3}$$

While learning parameters from documents, we use the output y_t of $\boldsymbol{x}_t$ as the prediction the next word $\boldsymbol{x}_{t+1}$, and the object function is given in Eq. 4.

$$\min \sum_{t=1}^{T-1} \left(\boldsymbol{y}_t - \boldsymbol{x}_{t+1}\right)^2 + \lambda \left(\sum_{i=1}^{5} ||\boldsymbol{W}^i||_2 + \sum_{j=1}^{4} ||\boldsymbol{U}^j||_2 \right) \tag{4}$$

Where $\sum_{t=1}^{T-1} \left(\boldsymbol{y}_t - \boldsymbol{x}_{t+1}\right)^2$ is the prediction loss, $\sum_{i=1}^{5} ||\boldsymbol{W}^i||_2 + \sum_{j=1}^{4} ||\boldsymbol{U}^j||_2$ is the regularization item, λ is the weight of the

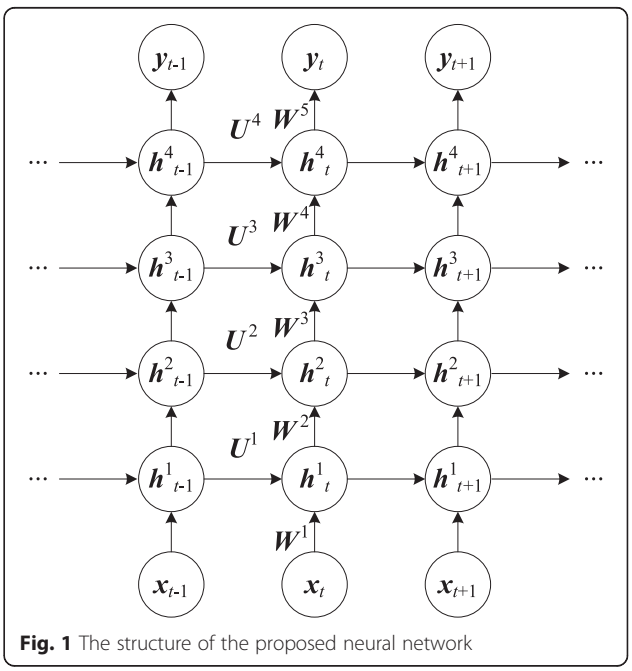

Fig. 1 The structure of the proposed neural network

regularization item, and $||W||_2$ is the sum of all squared elements in W.

With the document collection D, we can train the above model until stabilization and then get an embedded vector for each document. For each document d, we input x_t ($t = 1, ..., T$) in turn, compute all latent vectors, and use h_T^4 to represent the embedded vector of d, i.e., $v_d = h_T^4$.

1.1.2.2 Adding memory to neural cell For each neural cell of the above neural network, we apply long short-term memory (LSTM) [15] to represent the backward dependences between words. The structure of LSTM is in Fig. 2, and the computation for each unit is as follows.

$$i_t = \sigma(W_i x_t + U_i h_{t-1}) \tag{5}$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1}) \tag{6}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1}) \tag{7}$$

$$m_t = f_t \odot m_{t-1} + i_t \odot \sigma(W_m x_t + U_m h_{t-1}) \tag{8}$$

$$h_t = o_t \odot \sigma(m_t) \tag{9}$$

In the above equations, the meaning of Sigmoid function $\sigma(\cdot)$ is the same as before and $\odot$ is the dot product. Details of LSTM can be found in [15].

1.1.2.3 Adding feedback to recurrent neural network LSTM-based recurrent neural networks can only capture backward dependences between words of a document. In order to capture forward dependences between words, we need to add feedback links between neural cells. Figure 3 illustrates our proposed feedback recurrent neural network, and this model can capture the relationships between x_t with x_{t+1} and x_{t+2}.

In Fig. 3, each dotted line is a time slot, and the computation of h_t^1 and h_t^2 is the same as Fig. 1. However, the computation of h_t^3 and h_t^4 depends on three neural cells of the last time slot and can be computed as follows.

$$h_t^i = \frac{1}{1 + \exp\left(-U^i h_{t-1}^i - W^i h_t^{i-1} - W^{i+3} h_{t+1}^{i-2}\right)}, i = 3, 4 \tag{10}$$

In the proposed four-latent-layer feedback recurrent neural network, we can capture the relationships between x_t with x_{t+1} and x_{t+2}. If the number of latent layers is k, then we can capture the relationships between x_t with its following $k-2$ words. So, the proposed feedback recurrent neural network can capture forward dependences between words.

1.1.3 Embedded vector-based topic model

From Section 1.1, we know that each document $d \in D$ can be denoted with a vector v_d ($v_d = h_T^4$), where the dimension of v_d is p. Given d, we randomly choose a word w from d, substitute w with a random word $w' \neq w$ and then get a muted document d'. For d and d', the only difference is between w with w', so we assume that they have similar topics. For choosing the muted word w', we choose synonyms at the beginning, but the result is not

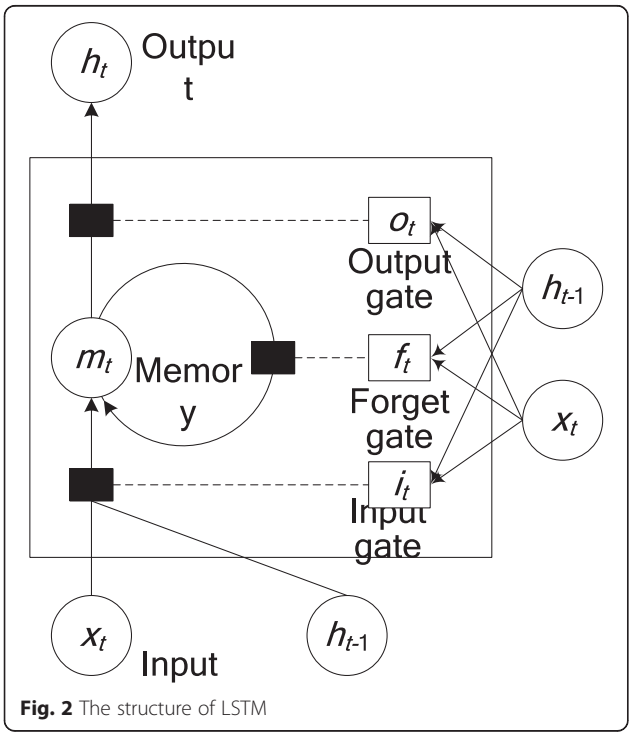

Fig. 2 The structure of LSTM

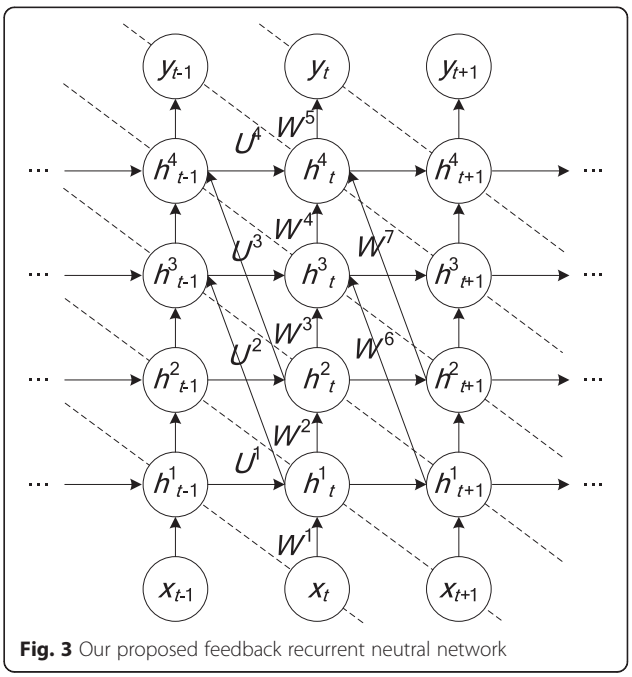

Fig. 3 Our proposed feedback recurrent neutral network

very well. The reason is that by substituting a word with its synonym, the two documents are almost the same, so the generalization of the model is very weak.

After embedding d and d' into continuous vector space, we get $\boldsymbol{v}_d$ and $\boldsymbol{v}'_d$. Then, we use the topic model described in Fig. 4 to analyze the underlying topics in a document collection. We assume that there are k topics in the document collection. Taking $\boldsymbol{v}_d$ and $\boldsymbol{v}'_d$ as input, if the probability that both d and d' belong to i-th topic is z_i, then we can estimate it as follows.

$$\hat{z}_i = \frac{1}{1 + \exp(-\boldsymbol{W}_i \boldsymbol{v}_d \boldsymbol{v}'_d)}, i = 1, ..., k \tag{11}$$

where $\boldsymbol{W}_i$ is a $p \times p$ matrix, and the whole $\boldsymbol{W}$ is a $p \times p \times k$ tensor.

While training the above model with d and d' pairs, if d' is a muted document of d, then $z_i = 1$; and if d' is a randomly chosen document except for d, then $z_i = 0$. During learning parameters of the model, we apply the sum of squared errors as loss function and L_2 regularization, then the objective is

$$\min \sum_{(d,d')} \sum_{i=1}^{k} \left(z_i^{(d,d')} - z_i^{(d,d')} \right)^2 + \lambda \sum_{j=1}^{k} ||\boldsymbol{W}_j||_2 \tag{12}$$

1.1.4 Experiments

The experimental platform is a laptop with Intel Core i5-2450 2.5GHz CPU and 4GB memory. The operation system is Ubuntu 10.04, and all algorithms are implemented with Java.

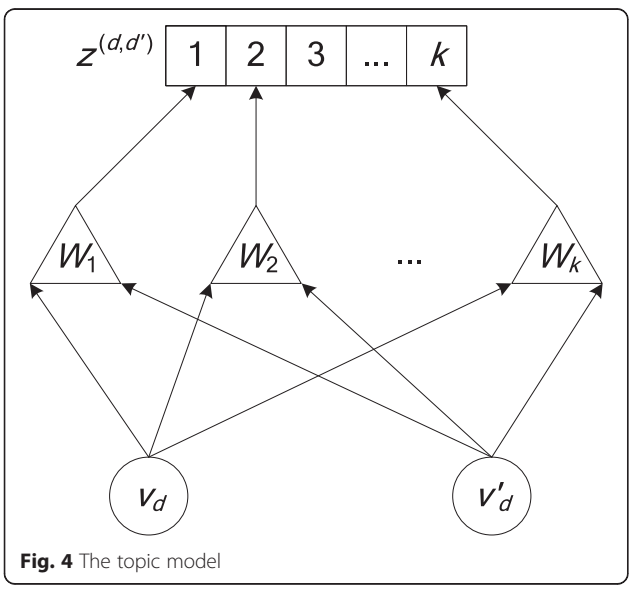

Fig. 4 The topic model

Table 1 Statistics of datasets

	NIPS	RML
W	13,649	16,994
D	1740	19,813
T	348	6188
N	23.0 M	1.27 M

1.1.5 Datasets

In the experiments, we use NIPS and RML two public datasets from [16], and the statistics of datasets is in Table 1. In Table 1, W is the number of distinct words after removing stop words, D is the number of documents including training and test data, T is the number of documents in test data, and N is the total number of words after removing stop words.

1.1.6 Baseline algorithms

We denote our proposed topic model with GRNN and compare it with PLSA [4], HDP-LDA [7], seTF-IDF [6], and DSSM [13]. PLSA is the basic probabilistic latent semantic analysis method, HDP-LDA extends LDA by modeling topics with hierarchical Dirichlet generation process, seTF-IDF is a term frequency—inverse document frequency method that takes semantics into consideration, and DSSM analyzes semantics by denoting each document with a set of triple characters.

1.1.7 Metrics

We apply point-wise mutual information (PMI) [17] and perplexity [18] to measure the performance of algorithms. PMI is used to measure the mutual information between words of a topic, and its computation is in Eq. 13. Bigger PMI means a better algorithm. Perplexity is used to measure the energy of entropy in a

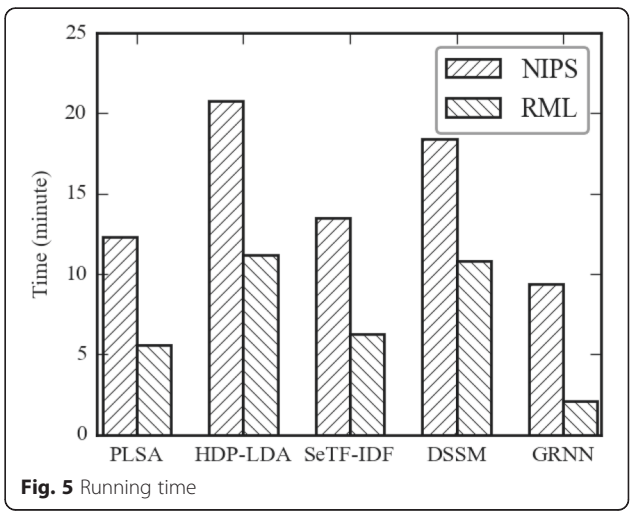

Fig. 5 Running time

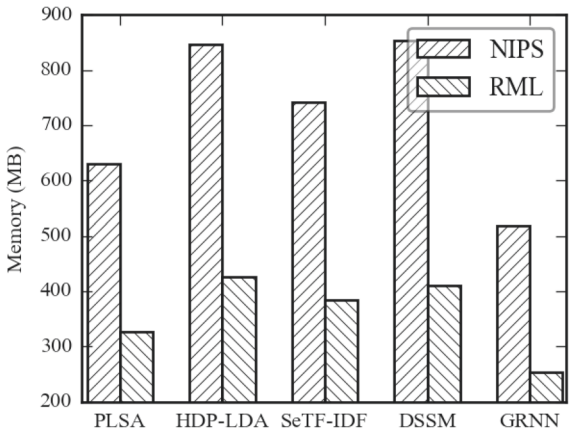

Fig. 6 Maximum memory usage of algorithm

topic, and smaller is better. Computation of perplexity is in Eq. 14.

$$\mathrm{PMI}(x,y) = \log_2 \frac{P(x,y)}{P(x)P(y)} \tag{13}$$

$$\mathrm{Perplexity}(k) = 2^{-\sum_{d\in T_k} \frac{1}{|T_k|} \log_2 p(d)} \tag{14}$$

Comparison of efficiency We compared running time and maximum memory usage of algorithms under the two datasets, and the results are in Figs. 5 and 6 respectively. In the comparison of running time, GRNN is obviously better than other algorithms. Running time of PLSA, HDP-LDA, SeTF-IDF and DSSM under RML dataset is one half of that under NIPS, and running time of GRNN under RML is a quarter of that under NIPS. For maximum memory usage under the two datasets, GRNN use 520MB and 230MB respectively, and both of them are smaller than other algorithms.

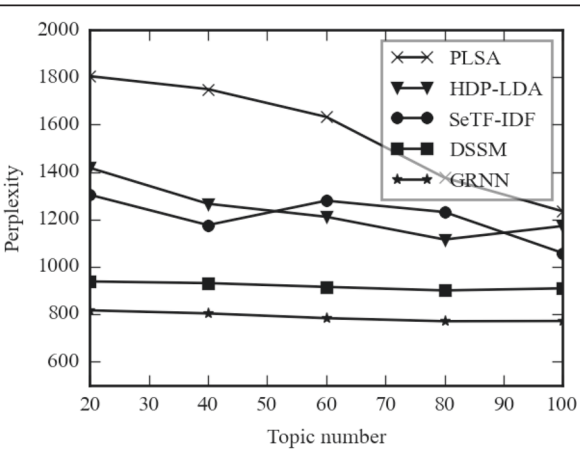

Fig. 7 Perplexity of PLSA and HDP-LDA

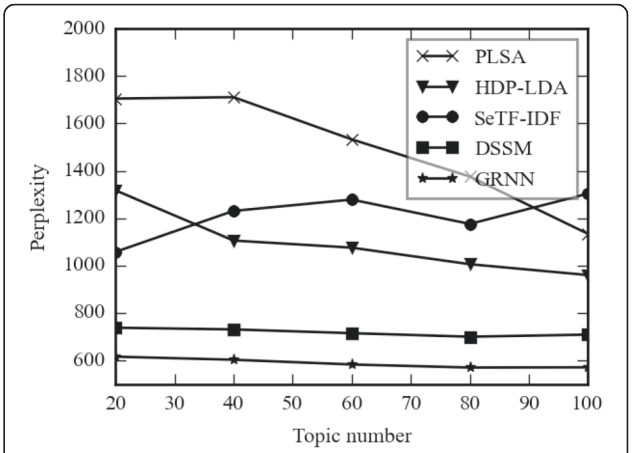

Fig. 8 Comparison of perplexity of HDP-LDA

1.1.8 Comparison of perplexity

By increasing the numbers of topics in both datasets, we compared the Perplexity of algorithms, and the results were in Figs. 7 and 8. In these two figures, Perplexities of PLSA and HDP-LDA decrease, Perplexity of SeTF-IDF does not have a trend, and Perplexities of DSSM and GRNN almost do not change. GRNN has the lowest Perplexity, DSSM has the second lowest Perplexity, and both of them are better than others.

1.1.9 Comparison of PMI

The same as comparison of Perplexity, we compared the PMI of algorithms by increasing the numbers of topics in both datasets, and the results were in Figs. 9 and 10. With the increasing of topic number, PMIs of all algorithms increase. In the NIPS dataset, when the number of topic is smaller than 45, PMI of DSSM is the biggest; and when the number of topic is bigger than 45, PMI of GRNN is the biggest. In the RML dataset, PMI of GRNN is bigger than others all the time.

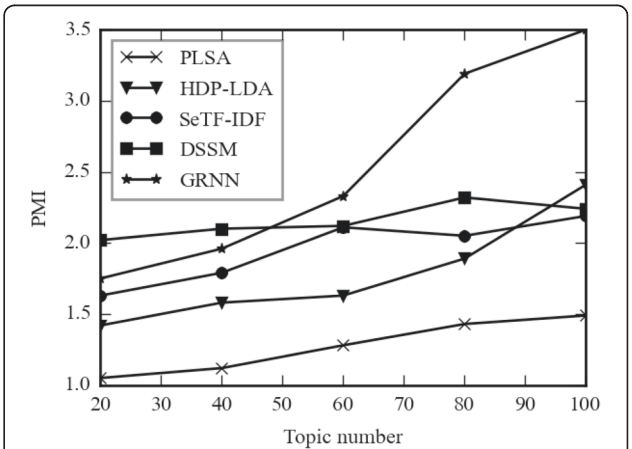

Fig. 9 Results of comparison of PMI

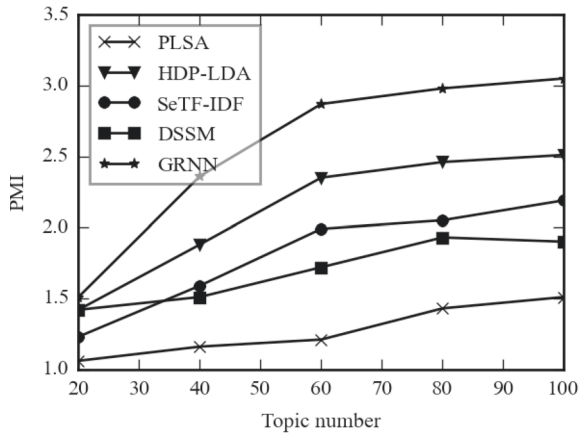

Fig. 10 PMI comparison under RML

2 Conclusions

In this paper, we proposed a feedback recurrent neural network to embed documents into continuous vector space and an embedded vector-based topic model. We applied long short-term memory recurrent neural network to capture backward dependences between words and feedback links between neural cells to capture forward dependences between words. With our proposed model, we can capture the relationships between the target word with its two following words. Massive experiments validate the effectiveness and efficiency of the proposed model.

Acknowledgements

The work was supported by the following funds: Science Research Foundation of Hunan Province Education Department(14C0483), Science Research Foundation for Distinguished Young Scholars of Hunan Province Education Department(14B070), and Science and Technology Project of Hunan Province of China(2014FJ6095).

Competing interests

The authors declare that they have no competing interests.

Author details

[1]School of Electronics and Information Engineering, Hunan University of Science and Engineering, Yongzhou, Hunan 425100, China. [2]School of computer sciences Chengdu Normal University, Sichuan, Chengdu 611130, China.

Received: 27 December 2015 Accepted: 8 June 2016
Published online: 16 July 2016

References

1. CD Manning, H Schutze, *Foundations of statistical natural language processing* (MIT press, Cambridge, MA, 1999)
2. DM Blei, Probabilistic topic models. Communications of the ACM **55**(4), 77–84 (2012)
3. N Evangelopoulos, X Zhang, VR Prybutok, Latent semantic analysis: five methodological recommendations. European Journal of Information Systems **21**(1), 70–86 (2012)
4. F Zhuang, G Karypis, X Ning et al., Multi-view learning via probabilistic latent semantic analysis. Information Sciences **199**, 20–30 (2012)
5. SP Crain, K Zhou, SH Yang et al., Dimensionality reduction and topic modeling: from latent semantic indexing to latent dirichlet allocation and beyond, in *Mining text data* (Springer, USA, 2012), pp. 129–161
6. A Aizawa, An information-theoretic perspective of tfidf measures. Information Processing & Management **39**(1), 45–65 (2003)
7. J Paisley, C Wang, DM Blei et al., Nested hierarchical Dirichlet processes. Pattern Analysis and Machine Intelligence, IEEE Transactions on **37**(2), 256–270 (2015)
8. A Bagheri, M Saraee, JF De, ADM-LDA: an aspect detection model based on topic modeling using the structure of review sentences. Journal of Information Science **40**(5), 621–636 (2014)
9. W. Ou, Z. Xie, X. Jia,B. Xie. Detection of topic communities in social networks based on Tri-LDA model, in *Proceedings of the 4th International Conference on Computer Engineering and Networks*. (Springer International Publishing, 2015), 1245–1253
10. C Zhang, J Sun, Large scale microblog mining using distributed MB-LDA, in *Proceedings of the 21st international conference companion on World Wide Web* (ACM, New York, 2012), pp. 1035–1042
11. T Mikolov, K Chen, G Corrado et al., Efficient estimation of word representations in vector space. arxiv preprint arxiv **1301**, 3781 (2013)
12. J Pennington, R Socher, CD Manning, Glove: global vectors for word representation. Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014) **12**, 1532–1543 (2014)
13. P Huang, X He, J Gao et al., Learning deep structured semantic models for web search using clickthrough data, in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management* (ACM, New York, 2013), pp. 2333–2338
14. K Du, MNS Swamy, Recurrent neural networks, in *Neural Networks and Statistical Learning* (Springer, London, 2014), pp. 337–353
15. S Hochreiter, J Schmidhuber, Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)
16. WL Buntine, S Mishra, Experiments with non-parametric topic models, in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (ACM, New York, 2014), pp. 881–890
17. D Newman, JH Lau, K Grieser, Automatic evaluation of topic coherence, in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (Association for Computational Linguistics, Stroudsburg, PA, USA, 2010), pp. 100–108
18. HM Wallach, I Murray, R Salakhutdinov et al., Evaluation methods for topic models, in *Proceedings of the 26th Annual International Conference on Machine Learning* (ACM, New York, 2009), pp. 1105–1112